ing your system editor (using Knuth's *TEX and* **METAFONT** as a guide), it is suggested that you might want your first attempt to be with the file TYPO.TEX. To run this example from the [TEX] directory (without using the startup command file provided) enter:

```
R TEX (Wait for the * prompt—
      this may take up to 20 seconds)
\input typo (Note lowercase.)
Watch the TEX comments—wait for system $ )
R LVSPOOL or R LHSPOOL
   (the system should handle the rest)
```

The mailing list for future informational memos regarding VAX/TEX will be the list of DUES PAYING TUG members who indicate a VAX interest, so make sure you join.

* * * * * * * * * * *

TEX FOR VAX/VMS

Barry Smith
Oregon Software

Well, it works—TEX for the VAX running VMS is alive, available, and in production use. (Production use is defined by example—we've just finished a 192 page manual for our optimizing PDP-11 Pascal compiler that is entirely typeset by TEX, including charts and diagrams.)

It's not yet perfect, nor something a "naive user" should be expected to enjoy. To list the major deficiencies:

- There are still annoying bugs, such as the flaming crash that occurs when one tries to insert a footnote. Most of these seem to be due to bugs in Digital's VAX Pascal, which just makes them harder to trace. (There's a new version due out soon, as always.)
- It's the "old" TEX-in-Pascal acquired from Stanford in late December. (Some of the bugs are real TEX bugs, too.) We've just received the recent official release, and as soon as I get a free weekend, ...
- There's absolutely no documentation whatsoever that relates to the VMS version. (Garçon, another weekend, please!) The amazing thing is that this doesn't seem to matter—Knuth's book describes the input format *exactly*, down to subtleties like tracing, and the installation directions are rather concise: "put everything in account [TEX]".

But, when it's good, it's very, very good—we're converting our skeptics to TEXnicians.

Details: TEX-in-Pascal for the VAX (11/750 and 11/780) running VMS, with about 50 Computer Modern fonts in assorted sizes. Comes with two spoolers (horizontal and vertical) for a Versatec 1200 printer/plotter, using the standard Versatec interface. Uses about 7000 blocks of disk space (mostly for fonts) and about a megabyte (whew!) of virtual memory while running. Comes in source and binary/executable forms, sources for spoolers, utilities, etc. (i.e., we'll send you everything we have). If you want to play with the Pascal programs, you'll want the fancy listings available from Stanford. A copy of the Pascal manual mentioned above will be included on request.

Supplied only on magnetic tape (600 foot) in "BCK/RST" format (that's an MCR utility for backup/restore, which works well for binary files). We can write 800 or 1600 b/in tapes—hearing nothing, we'll send 1600.

Fees: fifty dollars will get you a tape and shipment via UPS. To minimize our overhead, please don't send a tape, and do send a check or negotiable securities so we don't have to deal with purchase orders and billing (I'd prefer small unmarked bills). If you (or your friend) are in the "truly needy", just let me know.

Maintenance: We'll be working on bugs and convenience improvements for the perpetual future, and will be pleased to hear comments, suggestions, gripes, bugs—no promises of any response except through the TUGboat and Monte's newsletters.

* * * * * * * * * * *

Fonts

* * * * * * * * * * *

FONT COMMITTEE

Barry C. W. Doherty

Getting TEX up and running is only the first step in producing output. One needs also an output device with an appropriate driver and fonts. Transportable TEX output requires compatible fonts. Those sites which have similiar output devices (Varians etc.) and are willing to use the Stanford Computer Modern family have this problem solved.

There are many, however, who feel limited by the CM family as available from Stanford, either for reasons of design, completeness or merely preference for more traditional fonts (Helvetica, for instance).

In addition, many typesetters come with their own font libraries which people would like to use. A strong interest in these matters was shown by nearly everyone attending the TEX Implementors' Workshop. One result was the formation of a committee to investigate font-related issues. Members of the committee are

George Ball	Washington State University
Barry Doherty	American Mathematical Society
David Fuchs	Stanford University
Tom Hickey	OCLC
Ron Whitney	American Mathematical Society

It has been suggested that this committee should first try to identify vendors who are willing to work with the TEX community to provide font information (for TEX's metric files) and digitized representations of fonts (so that these fonts can be used with proof-quality devices), and to provide additional symbols for use with TEX. The role of vendors is complicated by licensing agreements, royalties, etc., and inter-vendor cooperation is unlikely. However, the TEX community may be large and strong enough to have the potential for affecting this development positively and to its benefit. It is probable, however, that such help from vendors will not produce fonts which can be reproduced on many other devices.

The need for a font library has also been pointed out, either one maintained centrally or with some central means for pointing people in the right direction. Many tools are already available for manipulating font files (e.g. translating them into editable form and repacking them). More such programs are needed, and they should be as transportable as TEX. (This could also help with the problem of disseminating font information, avoiding binary data entirely.) There is also a need for "device-independent" fonts, at least to the extent of uniform font metric files and a reasonable representation for proofing, so that a TEX file from an institution that had, say, Helvetica could have that file printed elsewhere.

Another role for the committee might be as a source for model RFP's to aid those institutions which must submit such documents to vendors; this would simplify the task of acquiring printers and fonts.

We welcome any ideas or comments.

* * * * * * * * * * *

THE STATUS OF METAFONT AT OCLC†

Thomas B. Hickey

We have been running METAFONT on Tandem hardware since August 1980. This was accomplished by recoding the SAIL version into T/TAL, the systems programming language on Tandem. The conversion took about six to eight weeks of concentrated effort plus another two to three weeks over nine months to add additional output modules and correct problems.

The system now is a full version of METAFONT, compatible with the original except in the following respects:

- TFM files are not yet generated
- All of an identifier's name contributes to its uniqueness, not just the first five characters plus length
- The limit on the size of **wxy** subscripts was lifted
- Use of <>, >= and <= for SAIL's one character relational symbols
- Limitations on the size of the raster (currently 300×300)
- Output routines adapted to drive Anadex printers

The conversion went fairly smoothly, the major problems being the lack of SAIL's nice strings and the difficulty in determining exactly what field in a word the SAIL version was using and what implicit initializations were being performed by shifts used to access and load fields.

Experience with METAFONT

In general, working with METAFONT is a pleasure. The interpretive nature of its execution allows a number of powerful tracing mechanisms which are very useful. In designing an alphabet the most difficult problems are deciding on the basic approach to be taken in specifying characters and writing the base routines which support such an approach. Once this is established, adding characters is fairly straightforward.

Most of the problems encountered in writing METAFONT programs arise from the complexities of the characters themselves, not METAFONT, but there are several features which users of other programming languages will occasionally miss:

- Loops
- Combinations of Boolean expressions
- A full complement of intrinsic functions
- Ability to write functions

†OCLC (Online Computer Library Center) is a not-for-profit computer library service and research organization based in Dublin, Ohio.

- Local variable hiding

Implicit in the design of METAFONT is the assumption that users have the ability to add their own output modules for specific devices. The METAFONT language offers no read access to the raster, so such routines must be written in the implementation language to be included in your METAFONT system. It is also possible, of course, to write programs which manipulate files in existing output modes, such as CHR, but this would not be a METAFONT program.

In my work in alphabet design I have found that much of the code is essentially defining two paths and filling in between them. Successors to METAFONT will undoubtedly include the ability to define and manipulate paths as entities. The concept of pens and erasers is certainly useful, but less fundamental than paths.

Conversion Problems

One of the reasons TAL was chosen for implementing METAFONT is its closeness in many ways to SAIL, but there were a number of problems:

- Lack of labelled case statements:
  This was overcome by setting up arrays of addresses for jump tables.
- A limit of two to nesting of procedures:
  Only a problem in the raster module. Overcome by moving and renaming variables.
- Lack of SAIL's strings:
  This necessitated the writing of a fairly comprehensive set of string handling routines.
- Lack of a macro facility as powerful as SAIL's:
  Some sort of macro facility would seem to be needed for a clean translation and TAL's was adequate for most purposes.
- Restrictions on file names:
  File names on the Tandem are limited to eight alphanumeric characters. Output file names are constructed by adding a single character to the front of them, so that CMR10 could produce files CCMR10, PCMR10, TCMR10, etc.
- 16 bit words:
  This probably had the greatest effect of all. The Tandem has an excellent repertoire of 32 bit arithmetic instructions but shifts and Boolean operations are only performed on 16 bit words. For ease of implementation and run time efficiency a 3 word 48 bit cell was used for dynamic memory to correspond with SAIL's 36 bit word. This does waste some space, especially in the double cell section (VMOM) of dynamic memory, but is quite efficient since shifts and masks are not required to access fields. In addition no new restrictions on sizes of fields in memory cells were needed.
- Different rounding of negative floating point numbers:
  Rounding is now performed by a function call.

Chel

I am presently undertaking the coding of an alphabet called Chel. This alphabet is based primarily on Helvetica and Helios and is designed to be usable over a broad range of widths and boldness. Chel is set up so that for each font to be generated a call to `Chelbegin` is made specifying point size, boldness and width. Boldness can have any of six values (extra light, light, regular, medium, bold, extra bold) and three width values (condensed, regular, extended). The `Chelbegin` routine then sets up the pens, `x` height, and other parameters which are used by the character routines. None of the routines assume that the vertical and horizontal raster resolution are equal.

The most important parameters are `efactor` and `bfactor` which control the expansion and boldness. If a user is not satisfied with the range of values generated by the standard widths and boldness these parameters can be controlled directly. The characters are designed so that in nearly all cases they change smoothly with `efactor` and `bfactor`. Exceptions to this include the tail on the lowercase 'a' which has an abrupt transition when the boldness is increased. I estimate that designing characters to work over such a broad range takes several times the effort that designing a single font would. Individual characters can be completed in as little as 15 minutes, or may take several days. An average character takes 3–4 hours.

Rough routines for both upper and lower case have been completed, but a great deal of work remains to be done to refine them and make them work correctly in all weights and widths. Figure 1 is a lower case 'b' in proofmode. Figure 2 shows this character in its 36 major variations (3 widths, 6 weights, and 2 slants). Both of these were printed on an Anadex 9501 printer. Completion date for this alphabet is somewhat uncertain although a useful version should be finished this summer. While it is anticipated that Chel will be made available outside OCLC, no distribution mechanism has been established.

The letter b
File created by METAFONT 06/05/81 08:38:29 AM
"The letter b"

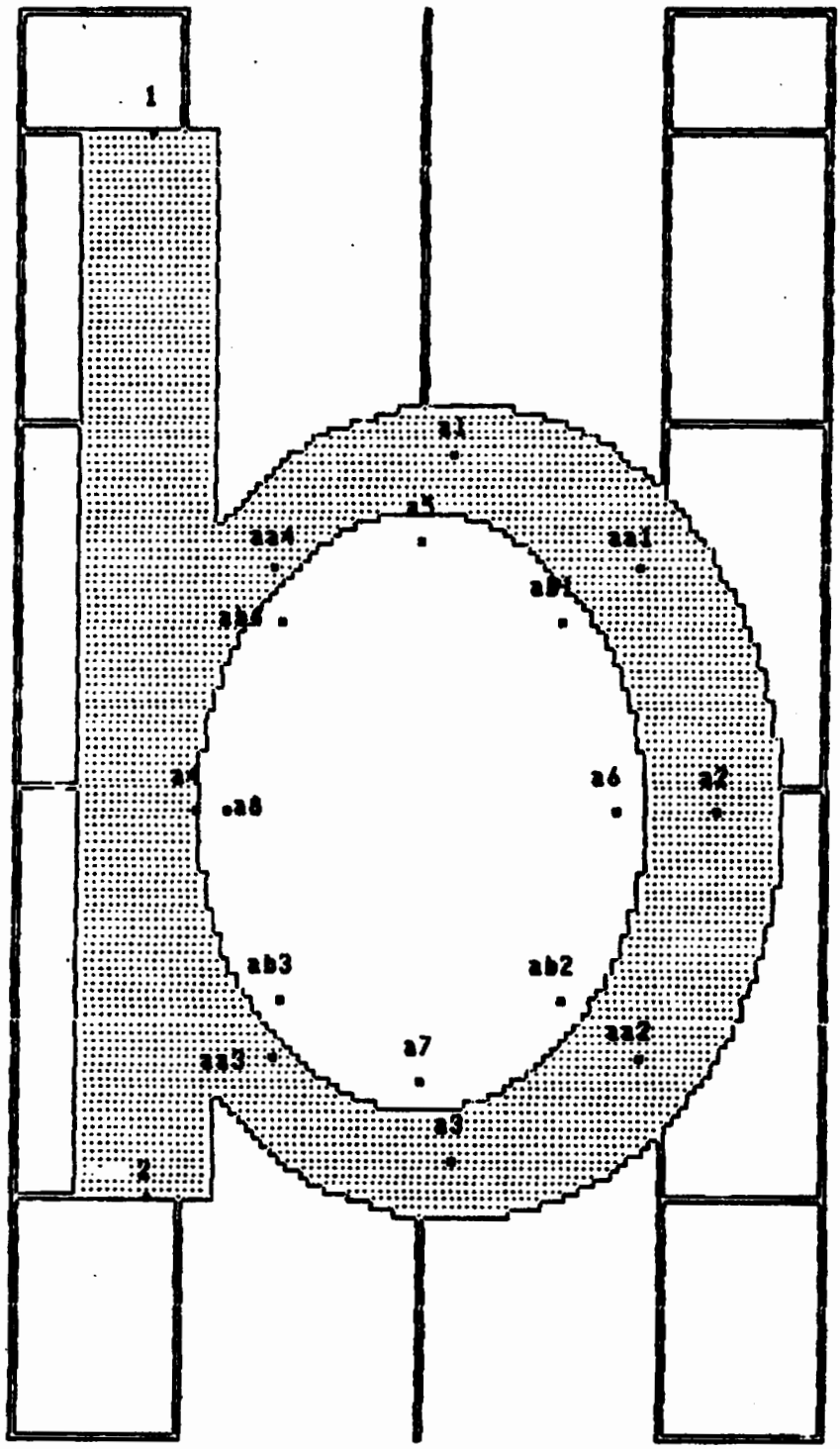


Figure 1. Lowercase "b": Proofmode at OCLC.

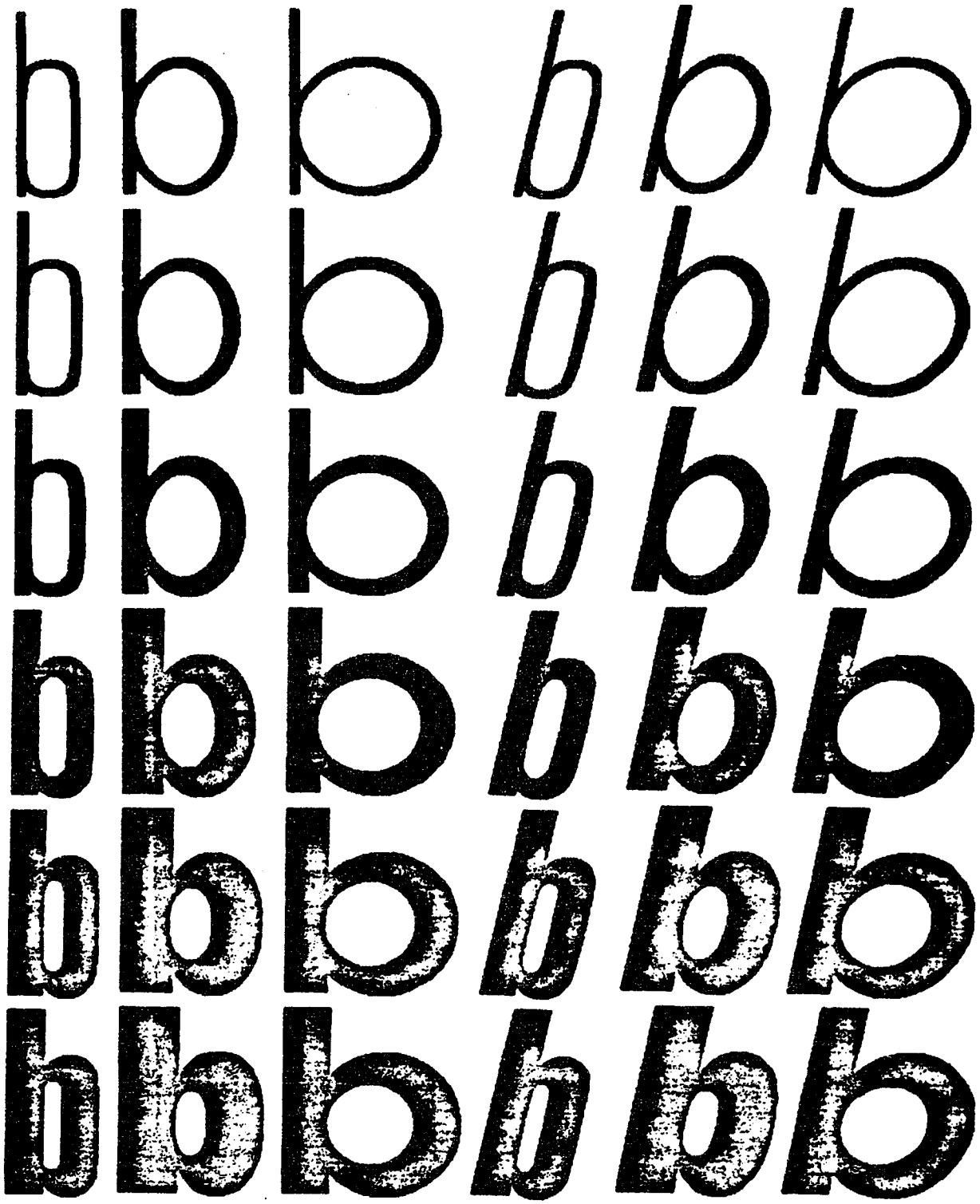


Figure 2. Variations on the letter "b": Chel font at OCLC.

FONT DEVELOPMENT AT THE AMERICAN MATHEMATICAL SOCIETY

William J. LeVeque

As Don Knuth has declared, his primary purpose in developing **METAFONT** and TEX was to be able to produce new editions of his book that would look like the originals. So it was reasonable that he should focus his attention, while designing particular alphabets and character sets in the Computer Modern family, on those fonts needed for composing ordinary text and mathematics. Roughly speaking, these were Computer Modern Roman, Bold Face, Slant, Sans Serif, and math mode Italics, Greek, symbols and script, in 5, 6, 7, 8, 9, and 10 point sizes, and Upright Italic and Typewriter in 9 and 10 point sizes.

When the American Mathematical Society (AMS) decided to continue the development of Knuth's system so that it could be used for the production of a wide variety of Society publications, it was apparent that additional fonts and symbols would be needed. Here are examples of some of the gaps we saw:

- Mathematical Reviews uses over 250 distinct non-alphabetic mathematical symbols, of which fewer than 100 are available from Knuth's font tables.
- There is a large number of diacritical marks which are used in the various languages that Mathematical Reviews quotes references in, and they occur on a larger number of letters than one might expect; it is not feasible to reserve a place in a font table for each combination of letter with diacritic. (And of course mathematicians put accents on letters in entirely unpredictable ways.). If the accents are designed separately, then the letters should be vertical if they are to center properly under or over the accents. This requires, for example, an upright Greek alphabet.
- Mathematical Reviews uses the whole Cyrillic family, none of which had been designed.
- Various publications such as the Combined Membership List are produced in very small type (5 or 6 point), and these had been designed by Knuth only for sub- and superscripts, not for text. What was needed was a text face as easy to read as that in a telephone directory.

The AMS Trustees decided to create a Font Committee (actually a subcommittee of its Committee on Composition Technology) to work on this problem; this committee contains both AMS staff members and working mathematicians who have had prior experience or interest in type design. Also, they commissioned Hermann Zapf, one of the world's outstanding type designers, to design several fonts especially for the AMS which would be esthetically consistent with Computer Modern Roman. It was agreed that the augmented font family, containing not only Knuth's and Zapf's fonts but also a large collection of mathematical symbols already designed by an AMS staff member, Mrs. Phoebe Murdock, for other purposes, should be called the Euler family.

Dr. Zapf has now prepared drawings of four alphabets: upright Greek and Fraktur (German), upright script, and an entirely new kind of alphabet which will be referred to as "handwritten". The latter is rather like an upright italic, and will be used in place of Knuth's math mode italics. Zapf's drawings have been distributed to the Font Committee, and the resulting suggestions have led to modifications of several of the characters, to accord better with what mathematicians are accustomed to.

The next step is to get **METAFONT** programs written, to simulate these designs in 10-point type. This work, now being done by one of Knuth's graduate students, Scott Kim, should be completed within a few weeks.

One of Knuth's **METAFONT** programs automatically converts a "roman" version of a character (whatever that means, for Greek or Fraktur) to four others: slanted, sans serif, bold and typewriter. So these five variants will immediately be available for these characters and alphabets.

There is an additional step—or rather, there are many hundreds of additional steps—required to yield what could be regarded as a complete collection of fonts in the Euler family. As is well known to type designers, type faces do not satisfactorily scale up or down in size in a linear way. A 10-point font reduced photographically to 6 points does not look right; hair lines tend to vanish altogether, thick lines get too thick, upper case letters begin to dominate small lower case letters, etc. So it is necessary to adjust individually the various parameters that govern the relative sizes of these features, for each desired size. This is not a terribly time-consuming job, for a single alphabet—but when there are five alphabets, in five variant forms, in ten or fifteen sizes, plus all the mathematical symbols and their variations, the job looms large. This task is already under way in the Society office; it is expected to be about a year before all the font tables one could reasonably ask for are available, in the Euler family. But long before then, of course, the ones most urgently needed will have been completed, and it is hoped that TEX, and Michael Spivak's macro package, $\mathcal{A}\mathcal{M}\mathcal{S}$-TEX, can

be used for most of the composition work of the Society by the end of 1981.

At first these fonts will be for the Society's own use, but it is hoped that through licensing agreements the Euler family can be made generally available to other printing houses.

* * * * * * * * * * *

TPHON

Ronald Whitney
American Mathematical Society

As was mentioned in Bill LeVeque's article (see page 39), the AMS has been using **METAFONT** to generate a set of small, highly condensed fonts to be used in the *Combined Membership List* of the Society and two other mathematical organizations. We started by creating a 6 point sans-serif Computer Modern font, and narrowed and thinned it until parameter twiddling no longer yielded significant improvements. Since proofmode was not yet available here and pens at Varian resolution were 1–2 pixels wide, more proofing than usual was done with the Alphatype. Inter-letter spacing seemed to provide the most difficulty.

As can be expected when a font family is pushed very far in a direction in which it is not designed to go, it became necessary for our purposes to bifurcate some parameters (serif correction, unit width, one pen size) and to modify some drawing routines. The result is our first approximation to what we have called the "telephone book" fonts. The TUG membership lists in this issue have been printed in TPHON and we welcome comments and criticisms about its readability.

* * * * * * * * * * *

PROOFMODE AND MAGNIFICATION

Barry C. W. Doherty and Ronald F. Whitney
American Mathematical Society

A large part of the TEX effort at the AMS has been the development of a library of fonts adequate for our typesetting requirements, and for this we have been making extensive use of (the SAIL version of) METAFONT. So far this has taken four directions: (1) creation of fonts in larger point sizes (e.g., 14 and 18 point), (2) filling in holes (e.g., 8-point text italic), (3) production of a complete set of fonts for the Florida Data and (4) creation of a 'telephone book font' (cf. the article by Ron Whitney in this issue, p. 40).

All but (3) have necessitated changing the parameters in the .MF files, but only with the last have really significant changes had to be introduced, to compensate for the changes of parameters subjected to excessive shrinkage. While METAFONT has been able to generate the Florida Data fonts, it has not been able to cope with the rounding problems made more visible by the low resolution of the output; some improvement might have been had by modifying the .MF files—at the expense of inconsistency of fonts across devices—leaving as the only practical alternative the manipulation of the raster pattern via an intermediary file. Changing the font description would also violate Knuth's goal of designing to the highest resolution device, in our case the Alphatype CRS.

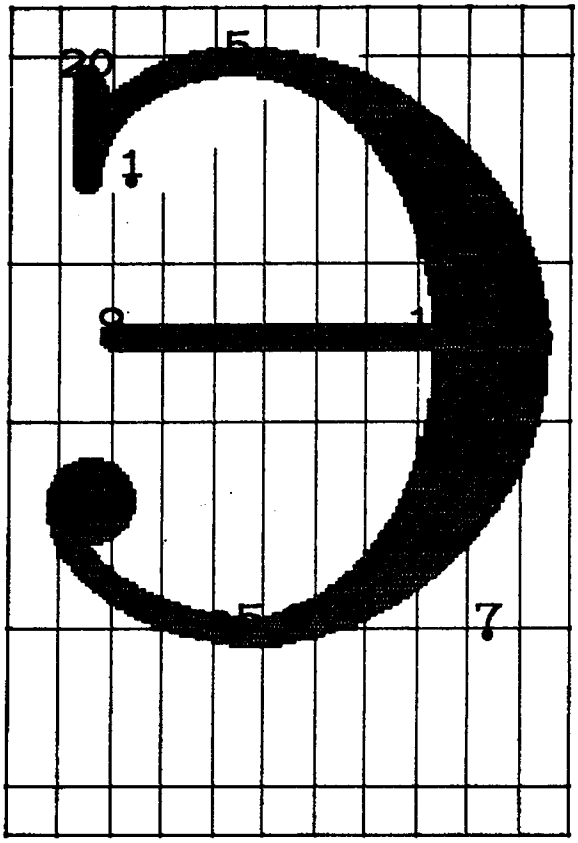
This development has been able to proceed without the availability of proofmode, although (4) concerns the creation of a new font, and would have been greatly facilitated had proofmode been available. Other pending work involves the creation of new fonts and new symbols. This work requires proofmode, or at least some means for examining an enlarged representation of the character.

Proofmode was implemented originally for the Xerox XGP. The routines for proofmode reside in the METAFONT module MFOUT. Our task was to rework those routines to produce proofmode on the Varian, which meant the translation of XGP commands to those suitable for the Varian (or the Florida Data), requiring the translation of absolute movement commands to relative ones. That is, for the XGP one could say 'move to column 545' whereas for the Varian one has to say 'move 224 units from last reference point'. The job was made more difficult by the fact that the XGP is not a available commercially, and it was not immediately clear how to compare the XGP with the Varian.

The basic procedure was to make METAFONT proofmode write a TEX-like .DVI file, containing all appropriate commands and typeset characters. Full utilization of the DVI commands PUSH and POP to save and restore one's current position on the page prevented the need for the introduction of several real parameters to keep track of where one is on the page (and the amount by which this position had changed). This also prevented the occurrence of numerous (accumulative) rounding errors.

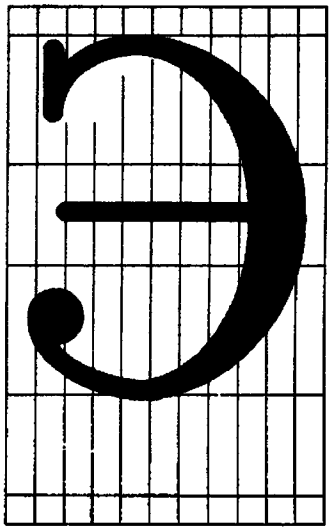
Changes were also necessary to the file-opening procedures to account for the fact that one was dealing with a binary rather than an ascii (text) file. At the same time the general file-handling procedures

Cyrillic uppercase Э.



Varian proofmode at the AMS.

Varian output of magnified character.



METAFONT character description:

```
"Cyrillic letter Eprime";
call charbegin(`037+3codeoffset,11,0,0,phh,0,rbowl);
cpen; top3y2=.3hh;
if bot3y2<.1hh: new y2; bot3y2=.1hh;
fi;
lft3x2=round .75u; w3 draw 2;                    % lower bulb
hpen; lft0x3=round 1.25u; y3=good6 .8hh; lft0x4=lft3x2; y4=y2;
x20=x3; rt4x3=rt0x1; y1=y3; top0y20=hh;
x5=.5[x3,x7]-u; x6=.5[x4,x7]; top0y5=hh+o; bot0y6=-o;
rt1x7=round(r-u); y7=y6;
if ucs$0: w0 ddraw 1..20, 3..20;                 % upper serif
        rpen#; w4 draw 3{0,1}..5{1,0};           % erase spurious part
fi;
hpen; w0 draw 3{0,1}..5{1,0};                    % shoulder
call `a darc(5,7,w5);                            % bowl
y9=y10=.52hh; x9=2u; x10=lft5x7;
w0 draw 9..10;                                   % bar
w0 draw 6{-1,0}..4{0,1}.                         % tail
```

were changed to to allow greater flexibility in both specifying and reporting the full file names.

Changes were worked out first on an 'archaic' TFX version of METAFONT but have since been transferred to a more recent (TFM) version in a rather straightforward manner. Some modifications in approach have been suggested by work done at Stanford, although the development has been independent.

A METAFONT description is device-independent as well as point-size-independent—within limits: that limit is METAFONT itself with its highly device-dependent routines and its inability to deal (linearly) with the full range of point sizes of interest to the typesetting community. The METAFONT description may be device independent but METAFONT is not, and that sort of generality is not possible without the use of translators to modify METAFONT's output. The consequence of this last is crucial with respect to the creation of a font library, and its implications are discussed elsewhere in this issue.

The larger figure on the previous page is a Cyrillic O produced by proofmode. Vertical unitlines and significant horizontal lines (representing h-height, x-height, axis-height, baseline, and descender depth, from top to bottom) form a grid on which the character is drawn. Points mentioned in the drawing routine are labelled.

In addition to proofmode (which takes work to implement), METAFONT contains a magnification option which is very easy to use. In ordinary font production for the Varian, one runs METAFONT and sets the variable `pixels` to 3.6. This value represents the number of pixels per point on that machine (increased by 30% since we use the Varian for proofing). To obtain the other character shown, one simply fools the Varian into thinking that one point (the printer's unit) contains 3.6*15 pixels. For the Stanford METAFONT this is done by specifying `mode=-1` (to tell METAFONT that you want a Varian font and that a magnification factor is coming) and `mag=15`. The same sort of procedure would work for any other output device for which METAFONT can produce characters.

Although significant points are neither indicated nor labelled with this magnification option, the figure obtained is of value in design. Subtleties of strokes are more likely to stand out here than in our Varian output of proofmode. It seems to us, for example, that the vertex at the bottom of the letter is more apparent in the smaller than in the larger figure.

* * * * * * * * * * * *

Warnings & Limitations

* * * * * * * * * * * *

Uppercase Update; Fickle Fonts

Last issue's warning about `\uppercase` has now been rendered obsolete by a change in TEX, whereby dimensions (`.5em`) and function words (`for`, `after`) are recognized in any combination of upper- and lower-case letters (`.5EM`, `For`, `aFtEr`). This change was made in Don Knuth's own version of TEX on February 27, 1981, and is described in the errata list among the "Extensions since last printing". But check the status of your local version before changing your macros!

Welcome though this change may be, it lays another trap: A macro for formatting entries in an index included the specification "... `\hangindent 10pt #1` ..." and an index entry "Forecasting" was rendered as "casting", indented just a bit more than the specified hanging indent. Don explains it thus:

"The word '`for`' is allowed after '`\hangindent`'; e.g.,

```
\hangindent 3pt for 5
```

and moreover you can use letters as constants as in

```
\char e     (=\char'145)
```

Then `\hangindent 3pt Forecasting` means `\hangindent 3pt for '145 casting`

Likewise, the word '`plus`' or '`minus`' will be gobbled after `\hskip 1pt ...`"

To disarm the trap, add a couple more braces to the offending macro: "... `\hangindent 10pt{}#1 ...`"

. . . . . .

Probably most sites use a version of TEX that has preloaded fonts, and probably most sites preload the fonts specified in `basic.tex` (see the TEX manual, Appendix B, for details). But some users, for whatever reason, decide to associate different fonts with the one-character names assumed by `basic`. When a file using non-`basic` font names is run through a version of TEX that has preloaded `basic`, the job may run to completion with no warning (the SAIL implementation of TEX gives no warning, but some Pascal versions may—at least the VAX/VMS version does so). TEX will use the metrics of the preloaded fonts, but the spooler will use the fonts requested by the user, and a ragged right margin may be only symptom.